Z-Ware Documentation

Release 0.1

Carsten Steffensen

Aug 29, 2018

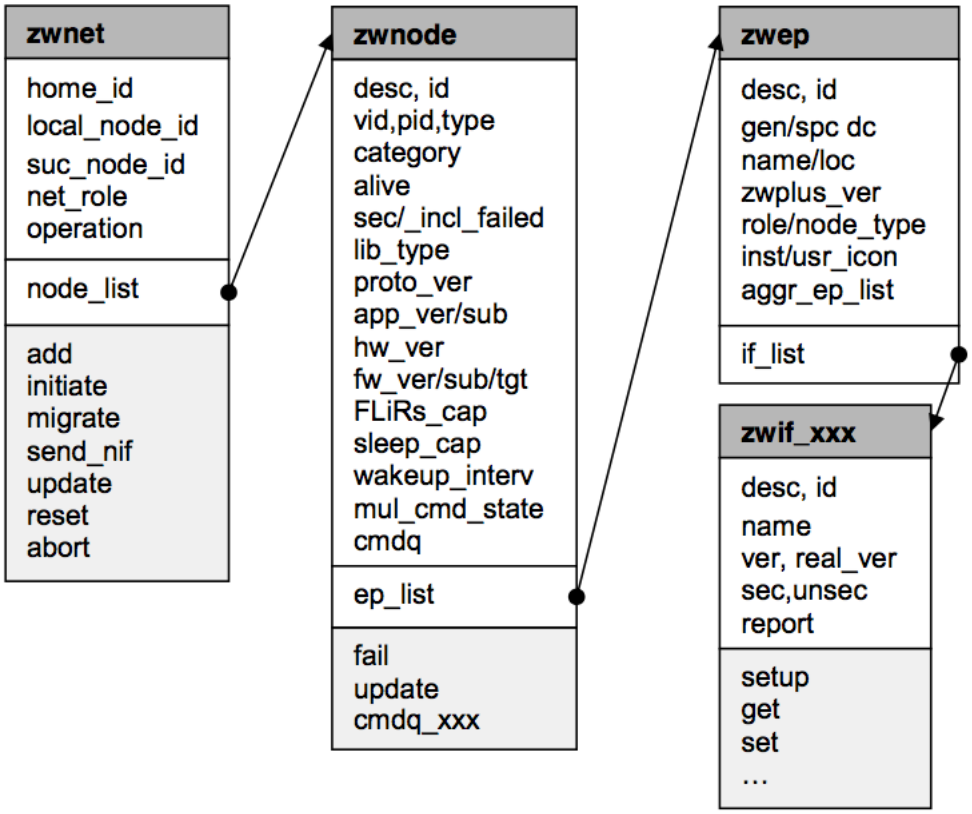
Z-Ware C-Lib API

Z-Ware is a Smart Home Gateway middleware designed for Z-Wave. Z-Ware makes Z-Wave integration simple by abstracts the Z-Wave network and devices into objects and exposes simple interfaces for application development.

CHAPTER 1

Z-Ware API structure

The Z-Ware API structure is as illustrated below. The application must loop through the structure to get a handle for the Network -> Nodes -> Endpoints -> Interfaces. The interfaces relates to the Z-Wave Command Classes. So after achieving the handle for the e.g. Binary Switch Interface in an Endpoint, it is possible to control the Binary Switch functions for the Endpoint.



The Z-Ware API may be accessed as either a WebAPI through HTTP POST commands

ZWNET
 The ZWNET object represents the Z-Wave network. Each Z-Ware application will only have one ZWNET instant through which network information and functions can be requested.

The application should start by connecting to the Z-Ware portal using the API's in [REGISTRATION API](https://documenter.getpostman.com/view/5175923/RWTrNc7Q#39f20a06-10d3-436b-aa1f-a0e41199192c). Once connected the application may request network information through the API's in [SYSTEM API](https://documenter.getpostman.com/view/5175923/RWTrNc7Q#cdc1fc65-5b3d-47bc-a290-e5984c4697e1) or control network functions through the API's in [NETWORK API](https://documenter.getpostman.com/view/5175923/RWTrNc7Q#a4c015b0-0432-45c5-b41a-302654e91f76).

The ZWNET object includes a node_list which is a list of the Z-Wave nodes in the network represented as ZWNODE objects. The node_list can be requested using the API [zwnet_get_node_list](https://documenter.getpostman.com/view/5175923/RWTrNc7Q#16e9600d-9f11-4aff-9963-19cc074772bf)

ZWNODE
 Each Z-Wave Node is represented by a ZWNODE object. The application may request node information or control node functions through the API's in [NODE/ENDPOINT API](https://documenter.getpostman.com/view/5175923/RWTrNc7Q#75310cdc-509c-4307-a844-d56ce3c28be9)

The ZWNODE object includes a ep_list which is a list of Node Endpoints supported by the node represented as ZWEP objects. The ep_list can be requested using the API [zwnode_get_ep_list](https://documenter.getpostman.com/view/5175923/RWTrNc7Q#d64bd835-f58d-4163-8ef2-7cd2074c79f8).

ZWEP
 Each Z-Wave Node Endpoint is represented by a ZWEP object. A Z-Wave node will as minimum have one Endpoint which represents the device. But some Z-Wave nodes may have more Endpoints like a power strip with multiple outlets. For this device each outlet will be represented as an Endpoint in addition to the device Endpoint.

Each ZWEP object includes a if_list which is a list of functionality related to the Endpoint represented as ZWIF objects. The if_list can be requested using the API [zwep_get_if_list](https://documenter.getpostman.com/view/5175923/RWTrNc7Q#06c28b31-93bb-43dc-88de-6cdc14512c6c)

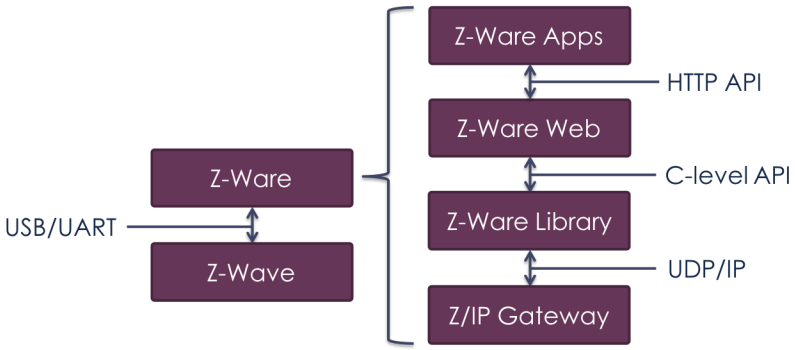
ZWIF
 Each functionality/interface in a Z-Wave Node Endpoint is represented by a ZWIF object. The interfaces relates to the Z-Wave Command Classes. So each Command Class supported by the Z-Wave Node Endpoint is represented as a ZWIF object.

CHAPTER 2

Overview

Z-Ware is a home control middleware that abstracts the Z-Wave network and devices into objects and exposes simple interfaces for application development.

Z-Ware is designed to run on Linux and will connect to the Z-Wave through UART or USB as shown in figure



The Z-Ware Library is a Z/IP (Z-Wave for Internet Protocol) Gateway (ZIPGW) client and abstracts Z-Wave Command Classes (CC) into controllable and monitorable interfaces over a C API (Application Programming Interface) for easier development and certification of Z-Wave controller applications. One such Z-Ware Library Client is the Z-Ware Web Server. The Library can be compiled into Consumer Electronics (CE) or Portal mode. In Portal mode, multiple ZIPGWs connect to the Library over TCP (Transmission Control Protocol) encrypted with TLS (Transport Layer Security) where the Library Client is expected to run in the Cloud. In CE mode, the Library connects to the ZIPGW via UDP (User Datagram Protocol) encrypted with DTLS (Datagram TLS) where the Library Client is expected to run in the home.

This document describes the Library usage and API with clear differentiation between CE & Portal modes where applicable.

CHAPTER 3

test

3.1 zwnet_init

Initialize network and return network handle

3.1.1 Declaration

func *int zwnet_init(const zwnet_init_p init, zwnet_p* net)*

```
int zwnet_init(const zwnet_init_p init, zwnet_p* net)
```

3.1.2 Parameters

[in] init User filled initialization information

[out] net Handle to network for use in other zwnet_xxx API calls

3.1.3 Returns

ZW_ERR_NONE on success; else ZW_ERR_XXX

3.1.4 Description

This call runs a state-machine to acquire the ZIPGW attached controller Home ID, controller's Node ID, HAN address and node list of the HAN. An internal network data structure is created and initialized with each of the node id found in the acquired node list. User application could get access to the controller Home ID and Node ID by calling zwnet_get_desc API only after the zwnet_notify_fn callback function returns status is ZW_ERR_NONE.

To populate the internal network data structure with endpoints and interfaces, this API tries to retrieve the node information from an internally maintained database. For those nodes found in the controller routing table but do not have corresponding node information in the database, the node info state-machine is invoked to get the information (CCs supported, CC version, node name & location, manufacturer id, product type id, product id and multi-instance/channel endpoints, etc) directly from the node device.

3.1.5 zwnet_init_p

bla bla bla